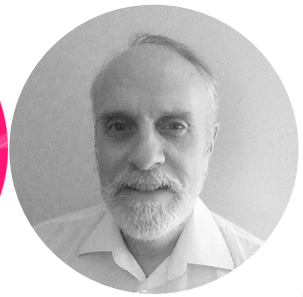
The Mediator Pattern



Gerald Britton

Pluralsight Author

@GeraldBritton www.linkedin.com/in/geraldbritton

Overview

Object-oriented design

- Single responsibility principle

Object interactions can multiply

- Program can begin to look monolithic

GUI application

- e.g. Visual Studio Code

More difficult to change the system

- Many objects depend upon each other
- Broken dependency inversion principle

Demo

Motivating example:

- Pet handler
- Cat, dog, fish

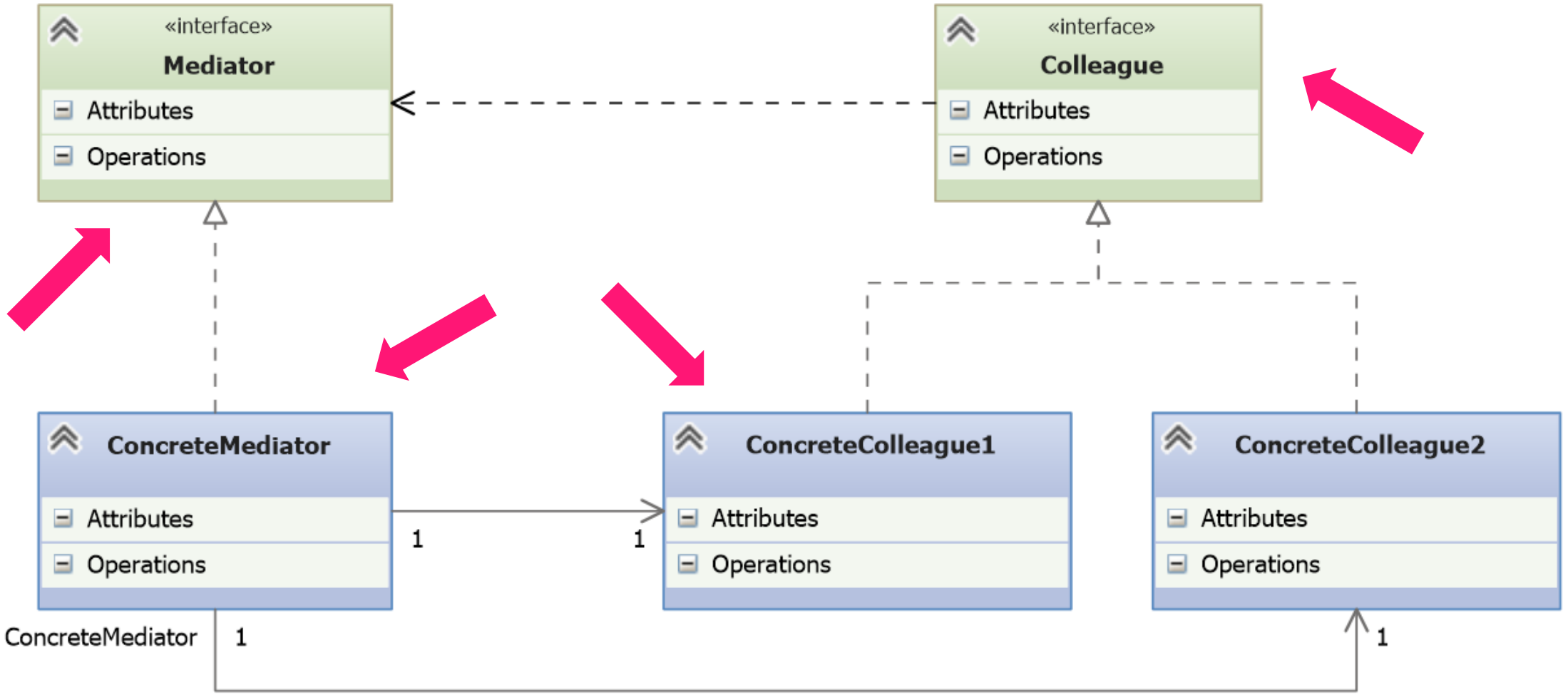
Mediator Applicability

Objects have many interdependencies

Hard to reuse objects with many references to others

Customize behavior without subclassing

Mediator Structure



Demo

Refactor with Mediator

Create a PetMediator class

Remove direct references between pets

Each pet will use the mediator instead

Implement time-of-day actions

Mediator Consequences

Benefits	Drawbacks
Reduces need for subclassing	Can become overly complex
Increases reusability by decoupling	Centralizes control
Simplifies maintenance	
Colleagues can vary independently	

Summary

- Mediator reduces colleague interactions
- Increases reusability
- Often used in GUI applications
- Can be complex
- Can become monolithic
- Many benefits to gain